Модел. и анализ информ. систем. Т. **21**, № **6** (2014) 155–168

УДК 519.178

Программный комплекс для анализа и визуализации графов

[1]Коломейченко М. И., [2]Золотых А. А., [1]Поляков И. В., [1]Чеповский А. М.

[1]*Национальный исследовательский университет "Высшая школа экономики"*
101000 Россия, г. Москва, ул. Мясницкая, 20
[2]*Московский государственный университет печати имени Ивана Федорова*
127550 Россия, г. Москва, ул. Прянишникова, 2А

e-mail: {maxim.kolomeychenko,azolotykh,igorp86}@mail.ru, achepovskiy@hse.ru

получена 20 сентября 2014

Ключевые слова: граф, анализ графа, визуализация графа, графовое хранилище, архитектура программного обеспечения

В данной работе представлено описание программного комплекса для хранения, анализа и визуализации графов социальных сетей. Проводится сравнительный анализ существующих программных продуктов для анализа и визуализации графов. Кроме того, представлена общая архитектура приложения, описаны принципы его построения и работы основных модулей. Отдельно приведено описание разработанного графового хранилища, ориентированного на хранение и обработку графов больших размеров. В качестве основной функциональности программного продукта представлены разработанный алгоритм выделения сообществ и реализованные алгоритмы авторазмещения графов. Преимущество разработанного программного продукта заключается в высокой скорости работы с сетями больших размеров, достигающих нескольких миллионов вершин и связей. Кроме того, используемая архитектура хранилища графов является уникальной и не имеет аналогов на данный момент. Имеющиеся в ней подходы и алгоритмы оптимизированы для работы с большими графами и обладают высокой производительностью.

Введение

При анализе графов социальных сетей возникает задача визуального представления их структур. Актуальность работы определяется потребностью в программном комплексе для анализа и визуализации графов больших размеров, возникающей в задачах социологического и маркетингового анализа.

Представленное программное обеспечение предоставляет широкие возможности для визуализации и анализа сетей больших размеров. Область применения обширна [1] и затрагивает многие смежные дисциплины, такие как социология, психология, политология, маркетинг и т.д.

Существует достаточно много промышленных продуктов для анализа графов. Например, i2 Analyst's Notebook [5], Sentinel Visualizer [6], CrimeLink [7], Xanalys Link Explorer [9] и Tom Sawyer Software [8]. Несмотря на некоторые различия в деталях, по предоставляемой функциональности и назначению эти системы визуализации во многом схожи.

i2 Analyst's Notebook, CrimeLink, Sentinel Visualizer и Xanalys Link Explorer являются программными продуктами, предназначенными для анализа систем взаимосвязанных объектов и изучения динамики последовательных событий. Tom Sawyer Software представляет собой набор библиотек для создания инструментов визуализации и анализа сетей из различных предметных областей.

У большинства современных промышленных систем присутствует ряд недостатков: отсутствие платформенно-независимых решений, отсутствие собственных специализированных хранилищ, системы не ориентированы на работу с графами больших размеров.

1. Архитектура программного комплекса

Разработанный программный комплекс предназначен для построения и анализа графов социальных сетей и автоматизированного размещения вершин и связей в соответствии с задаваемыми изобразительными соглашениями.

Согласно поставленной задаче и выбранным изобразительным соглашениям спроектированы эффективные структуры данных, которые позволили создать развитую систему редактирования и анализа без ущерба производительности и с допустимым объемом занимаемой памяти.

Ядро программного комплекса реализовано на языке C++ с использованием открытой графической библиотеки OpenGL для реализации двухмерной компьютерной графики и с использованием библиотеки Qt с открытым исходным кодом для реализации элементов графического интерфейса.

Теперь опишем общую работу и взаимодействие модулей разработанной архитектуры. Ядро приложения состоит из 6 основных модулей. Ниже приведено краткое описание основных модулей.

1. Модуль конвертации данных.

   Предназначен для импорта данных из различных внешних форматов в специализированный единый формат хранения.

2. Хранилище графов.

   Специально разработанное хранилище, позволяющее хранить графы больших размеров и предоставляющее эффективные методы по работе с ними.

3. Модуль алгоритмов анализа.

   Включает в себя разнообразные алгоритмы анализа графов.

4. Модуль алгоритмов авторазмещений.

   Реализует базовые размещения, позволяющие проанализировать структурные особенности графов.

5. Модуль визуализации.

   Реализует богатую функциональность по отображению различных элементов графов социальных сетей.

6. Хранилище визуализации графов.

   Позволяет хранить различные визуальные представления исследуемых графов.

Импорт данных в Хранилище и клиентское приложение осуществляется из Excel таблиц, файлов с расширениями: doc, html, xml, реляционной БД, или текстовых файлов специального формата, содержащих структурированные данные. Конвертер выделяет из файлов табличные данные и преобразовывает их в граф, который сохраняется в файловой системе хранилища. Методология работы с сетью предусматривает два режима хранения:

- базовое хранение, предназначенное для хранения графов большого размера (миллионы вершин и ребер);
- локальное хранение, предназначенное для хранения графов, являющихся подграфами, полученными запросами к базовым графам и результатами обработки клиентским приложением.

К каждому конкретному сохраненному графу может быть привязано любое количество вариантов визуализации локального графа, которые сохраняются в хранилищах вариантов визуализации. Под вариантом визуализации понимается расположение вершин и связей между собой, иконки всех вершин, способы визуализации всех вершин, иконки всех атрибутов и их стили, стили всех связей и координаты их точек изломов.

Модуль анализа сети предназначен для реализации алгоритмов и структуры данных и позволяет:

- Извлекать подграфы на основе поисков в ширину и на заданную в запросе глубину от заданной вершины, на основе используемых в запросах фильтров по атрибутам, на базе выделения кластеров.
- Осуществлять поиск объектов по заданным параметрам атрибутов в соответствии с различными фильтрами.
- Осуществлять поиск похожих вершин.
- Выполнять объединение и пересечения графов.

  Операции объединения и пересечения схожи по своей реализации. В качестве входных параметров для данных операций должны быть зафиксированы наборы атрибутов графа, с помощью которых будет проходить отождествление вершин и связей соответственно. По умолчанию учитываются все атрибуты.

  При выполнении операции пересечения строится новый граф, в котором множество вершин состоит из отождествленных вершин обоих графов по заранее заданному набору атрибутов. Множество связей нового графа состоит из

всех связей обоих графов, участвующих в операции. Затем для всех связей в новом графе между всеми парами вершин происходит операция фильтрации повторяющихся связей по заданному набору атрибутов. Такой подход позволяет пересекать графы с количеством вершин порядка нескольких миллионов и количеством связей порядка нескольких миллиардов в каждом. Алгоритм способен обрабатывать объемы данных, не помещающиеся в оперативную память, за счет последовательного считывания данных с жесткого диска при выполнении фильтрации дублирующихся связей.

Операция объединения двух графов аналогична пересечению всего лишь с одним отличием: множества вершин исходных графов объединяются, при этом отождествление вершин по набору атрибутов гарантирует отсутствие повторяющихся вершин в новом графе.

- Автоматически выделять сообщества.

При работе с небольшими графами (несколько десятков вершин) еще возможно выполнить размещение элементов сети вручную. При увеличении размеров исследуемого графа начиная с нескольких сотен вершин задача ручного размещения становится невыполнимой. В связи с большим разнообразием видов графов, существует множество различных способов автоматического отображения графов. Модуль авторазмещения предназначен для автоматического размещения объектов и реализует графические схемы [2, 3] типа павлиний хвост, линия темы, круговое размещение, покомпонентное круговое размещение, а также размещение, основанное на алгоритме выделения сообществ в сети. Все разработанные алгоритмы авторазмещений направлены на работу с большими графами, достигающими сотен тысяч вершин. Визуальное представление сетей может выступать в качестве мощного метода отображения скрытой информации, хранящейся в данных о системе взаимосвязанных объектов.

Модуль визуализации отвечает за способы отображения объектов сети и за скорость их отображения. В программном комплексе предусмотрено несколько изобразительных соглашений для вершин: иконки, линии темы, таблицы. Отдельно стоит отметить способ представления вершин в виде линии темы. Линия темы – это вершина, которая представляется в виде горизонтального отрезка с иконкой, причем все смежные вершины соединены с линией темы перпендикулярно (если они находятся над или под ней) или с ближайшим ее концом. Положение концов линии темы и ее длина могут меняться. Использование данного способа представления позволяет эффективно решать задачу визуализации для некоторых графов. Доступна визуализация множественных связей с изломами между вершинами, с возможностью задания ширины, цвета и типа линии. Для атрибутов также предусмотрено множество способов и настроек отображения.

Для эффективного отображения графов используются специально разработанные структуры данных, позволяющие получать быстрый доступ к данным, и алгоритмы машинной графики, заточенные под одновременное отображение большого количества объектов.

Все вершины и связи графа хранятся в соответствующих сбалансированных бинарных деревьях поиска, где ключами выступают указатели на объекты. Структура связей графа также хранится в сбалансированном бинарном дереве поиска, где в

качестве ключей выступают указатели на вершины, а в качестве значений хранятся множества инцидентных ребер соответствующей вершины, которые в свою очередь также реализованы на сбалансированных бинарных деревьях поиска. Выбор данной структуры обусловлен тем, что доступ к случайной вершине или связи, доступ к инцидентным связям вершины и удаление или добавление объектов будет осуществляться за логарифмическое время. При визуализации используются координаты вершин, хранящихся в структуре вершины. В то время как координаты связей вычисляются в процессе визуализации. Для этого предусмотрена отдельная структура данных, отвечающая за хранение направляющей линии между парой вершин. Все связи между заданной парой вершин будут строго параллельны направляющей линии и содержать такое же количество изломов. Тем самым достигается эффективное расходование оперативной памяти без ущерба производительности.

Процесс визуализации распараллелен на несколько уровней. Процесс отрисовки связей, вершин и атрибутов происходит параллельно, и только после полного выполнения всех потоков граф отображается на экране. Отдельно стоит отметить, что при перемещении или изменении одного или нескольких объектов схемы, перерисовывается не вся схема целиком, а только ее небольшая часть. Это достигается за счет применения описанной выше методики независимо для объектов двух типов: объектов, которые могут менять свое состояние в процессе выполнения операции, и объектов, остающихся неизменными в процессе выполнения операции. Практически большую часть времени объектов второго типа на порядки меньше, чем объектов первого типа, тем самым в процессе работы с графом часто выполняется перерисовка объектов второго типа, в то время как затратная операция перерисовки большого количества объектов первого типа выполняется заметно реже. За счет этого достигается высокая скорость визуализации графа и манипулирования данными.

2. Хранилище графов

Хранилище графов ориентировано главным образом на поддержку операций пополнения, объединения и пересечения графов, и поиска кратчайших путей между двумя группами вершин.

Используемые структуры данных рассчитаны на работу с графами, содержащими до 100 миллионов вершин и нескольких миллиардов связей, поэтому хранение графа в виде матрицы смежности не представляется возможным. За основу взят способ хранения графа в виде списков смежности.

В качестве базовой структуры хранилища используется специализированная файловая система. Данная система реализуется внутри некоторого файла стандартной файловой системы компьютера, поэтому под логическим файлом будем понимать файл, принадлежащий данной специализированной системе. Предполагается, что каждый логический файл состоит из некоторых записей переменной длины. Главные преимущества от использования собственной файловой системы состоят в следующем:

1. Отсутствие системных вызовов при работе с собственной файловой системой.

2. В традиционных файловых системах имеется существенное ограничение на

количество одновременно открытых файлов, которое отсутствует при использовании собственной файловой системы.

3. Управление буферизацией и хранением данных прозрачно и не зависит от особенностей операционной системы.

4. Отсутствие сложной системы адресации (структуры каталогов) и лишних механизмов (права доступа) приводит к отсутствию дополнительных накладных расходов на организацию хранения данных.

Все добавляемые в граф вершины идентифицируются возрастающими натуральными числами, начиная с 1. Атрибуты вершин складываются в некоторый файл по порядку. Если атрибуты очередной вершины требуют для своего хранения S байт, то для них резервируется $1.2 * S + C$ байт, где C–параметр алгоритма, который выбирается в зависимости от параметров атрибутов того графа, который планируется хранить. Наличие дополнительного места позволяет в дальнейшем изменять атрибуты некоторой вершины, в том числе путем добавления новых. Если свободного места недостаточно, для модифицируемой вершины будет выделена еще одна запись и обе записи будут объединены в двунаправленный список. В дальнейшем возможно выделение дополнительных записей. При этом возникает некоторая фрагментация, которая может быть устранена путем перестроения файла атрибутов вершин, что потребует полной остановки работы хранилища.

Каждой вершине v сопоставлен один логический файл $F(v)$, в котором хранятся записи, описывающие связи этой вершины. При этом один логический файл может быть сопоставлен нескольким вершинам.

Каждая связь l, добавляемая в хранилище, формирует некоторую запись, которая содержит идентификаторы двух соединяемых вершин, два бита характеризующие ее направленность (связь может быть ненаправленной), а также все атрибуты связи. Кроме того, каждой связи присваивается некоторый уникальный идентификатор, который также добавляется в виде отдельного поля к добавляемой записи.

При работе с хранилищем может возникнуть необходимость в полном удалении или изменении содержащихся в вершине или связи атрибутов.Соответствующие операции реализованы в хранилище.

Для более быстрого выполнения операций поиска вершин, по конкретным значениям атрибутов в хранилище присутствует специализированный индекс по атрибутам вершин. Данный индекс также использует собственную копию специализированной файловой системы.

Также реализована операция поиска кратчайших путей между парой объектов. Рассматриваемый объем данных делает невозможным использование стандартного алгоритма поиска в ширину для нахождения кратчайшего пути. Для реализации операций поиска пути и нахождения компонент связности используется сбалансированное дерево кортежей (a,b,N), состоящих из идентификаторов двух вершин и числа, равного количеству связей в графе между ними. Данная структура за логарифм от числа уникальных ребер операций позволяет получить все вершины смежные с данной с условием, что кратность связи должна быть выше заданного порога. С помощью этой структуры и поиска в ширину происходит поиск путей и обнаружение компонент связности.

Более сложные алгоритмы предполагают создание и поддержание дополнительных структур данных, размер которых потенциально может превосходить размер самого графа в несколько раз.

Используемая схема кластеризации списков смежности при построении графа не использует дополнительных структур данных, позволяя при этом значительно сократить количество дисковых операций, требуемых для нахождения какого-либо пути между двумя группами вершин, по сравнению с тривиальным алгоритмом поиска в ширину.

Используемые структуры данных позволяют проводить эффективное объединение и пересечение графов. Есть лишь два требования к хранилищам, необходимые для успешного выполнения этих операций:

Во-первых, списки вершин обоих хранилищ со всеми их атрибутами должны одновременно помещаться в оперативной памяти. Во-вторых, объединенное хранилище не должно содержать вершины, все связи которой вместе с атрибутами не помещались бы в оперативную память. Данная структура хранилища и предусмотренные операции были протестированы на графах, имеющих порядка 100 миллионов вершин и нескольких миллиардов связей.

3. Функциональность программного комплекса

Данный программный комплекс предоставляет широкую функциональность при работе с графами больших размеров.

В программном комплексе предусмотрено несколько изобразительных соглашений для вершин: иконки, линии темы, таблицы. Доступна визуализация множественных связей с изломами между вершинами, с возможностью задания ширины, цвета и типа линии. Для атрибутов также предусмотрено множество способов и настроек отображения.

Помимо богатой функциональности отображения различных элементов графа, разработанное ядро визуализации позволяет манипулировать большими графами, размеры которых достигают миллиона вершин и нескольких миллионов связей, за счет специально разработанных и оптимизированных алгоритмов машинной графики.

За счет связки с графовым хранилищем удобно работать с огромными графами путем выгрузки подграфов по заданным фильтрам, объединения и пересечения графов. Стоит отметить, что за счет наличия модуля конвертации данных все графы хранятся в едином специализированном формате хранилища.

Аналитические инструменты позволяют проводить разнообразные исследования графа [4] и его структуры:

- Поиск и выделение вершин и связей по заданным атрибутам.
- Подсчет метрик центральности для всех вершин и связей.

  В теории графов центральность элемента (вершины или связи) определяет его значимость относительно других объектов. В программном продукте реализованы несколько метрик центральности: центральность по посредничеству, центральность по степени, центральность по близости.

Например, используется метрика связи – центральность по посредничеству (betweenness centrality): количество кратчайших путей между всеми парами вершин, проходящих через заданную связь.

Центральность по степени – это отношение количества связей определённого узла к общему количеству других узлов.

Центральность по близости выражает, насколько близко узел расположен к остальным узлам сети. Формально, нормализованная центральность по близости (1) выражается как отношение числа других узлов графа к сумме расстояний между определённым узлом и всеми другими. Если значение центральности по близости равно 1, это означает, что определённый узел связан со всеми другими узлами.

$$ClosCent(v) = \frac{N-1}{\sum_{u \in V} d(u,v)}, \tag{1}$$

где V – множество вершин графа, $N = |V|$ – количество всех вершин, v – вершина, для которой выполняется вычисление метрики, $d(u,v)$ – функция расстояния от вершины u до вершины v.

Кроме того, для поиска важных объектов с точки зрения структуры графа выполняется поиск мостов и точек сочленения. Точкой сочленения (или точкой артикуляции) называется такая вершина, удаление которой увеличивает число компонент связности. Поиск всех точек сочленения в графе выполняется на основе поиска в глубину.

Мостом называется такое ребро, удаление которого увеличивает число компонент связности. Неформально эта задача ставится следующим образом: требуется найти на заданной карте дорог все "важные" дороги, т.е. такие дороги, что удаление любой из них приведёт к исчезновению пути между какой-то парой городов. Поиск всех мостов в графе также реализуется на основе поиска в глубину.

- Реализован алгоритм поиска близнецов в графе по заданному набору фильтров.

- Также можно проанализировать кратчайшие пути между любой парой вершин в сети.

  Для нахождения кратчайших путей между парой вершин используется алгоритм Дейкстры.

- Подсчет коэффициента кластеризации.

  Плотность – это отношение числа имеющихся рёбер графа к максимально возможному количеству рёбер данного графа. Метрика плотности используется в первую очередь при сравнении графов одного размера или при сравнении графа с самим собой во времени. Коэффициент кластеризации узла (2):

$$CC(v) = \frac{t_v}{q_v(q_v-1)/2}, \tag{2}$$

где v – вершина, для которой выполняется подсчет коэффициента, q_v – количество связанных с ней вершин, t_v – количество связей между всеми ее соседями. Число t_v – суммарное число треугольников (циклов длины 3), прикрепленных к узлу v. $q_v * (q_v - 1)/2$ – максимально возможное число треугольников. Если все ближайшие соседи узла v взаимосвязаны, то $CC(v) = 1$. Когда между соседними узлами вершины нет связей (как в случае с деревьями), то $\mathrm{CC}(v) = 0$.

Тогда кластеризация всей сети определяется как (3):

$$CC(G) = 3 * \frac{N_\Delta}{N_\vee}, \tag{3}$$

где N_Δ – число всех треугольников в графе, $N_\vee$ – число связанных троек, где «связанная тройка» означает узел и два его ближайших соседа.

Когда коэффициент кластеризации высокий – это означает, что граф чрезвычайно плотно сгруппирован вокруг нескольких узлов; когда он низкий – это значит, что связи в графе относительно равномерно распространены среди всех узлов.

В реализуемом программном продукте предусмотрен подсчет распределения степеней вершин и связей. Также можно вычислить коэффициент ассортативного смешивания по степеням вершин (4). Термин «ассортативное смешивание» возник в социологии при изучении формирования супружеских пар [4]. Социологические исследования показали, что социальные сети друзей также формируются на основе наличия общих признаков: языка, расы, возраста, уровня образования и т. д. В таких случаях говорят, что рассматриваемые сети обладают свойством ассортативного смешивания. Однако, если сети формируются по антагонистическому принципу, то они в свою очередь будут обладать свойством диссортативного смешивания. Ассортативность является степенью тенденции вершин сети формировать связи с узлами с одним и тем же числом связей.

$$AM(G) = \frac{M \sum_{e \in E} k_1^e k_2^e - [\sum_{e \in E} k_1^e]^2}{M \sum_{e \in E} (k_1^e)^2 - [\sum_{e \in E} k_1^e]^2}, \tag{4}$$

где G – исследуемый граф, E – множество всех ребер графа, $M = |E|$ – количество всех ребер графа, e – ребро графа, k_1^e – количество связей первой вершины из двух концов связи e, k_2^e – количество связей второй вершины из двух концов связи e.

Данный коэффициент принимает значения от -1 до 1. Если вершины с большим числом связей связаны друг с другом, то значение коэффициента близко к 1, если вершины с большим числом связей связаны с вершинами с меньшим числом связей, то значение коэффициента близко к -1.

При работе с большими графами для анализа структуры реализован алгоритм оценки связности, который позволяет для заданного порога по числу связей между парой вершин разбить граф на компоненты связности, выполнить размещение объектов с учетом полученных компонент и показать списки всех вершин для каждой полученной компоненты.

Кроме того, реализован алгоритм автоматического выделения сообществ с возможностью визуализации графа на основе полученного разбиения на сообщества.

Распознавание структуры, скрытой в реальных социальных сетях, является ключевой задачей, решение которой необходимо для понимания организации сложных сетей. Сложные системы обычно организуются в блоки, которые имеют определенные роли и функции. В представлении сети такие блоки являются множеством вершин с высокой плотностью внутренних связей, в то время как связи между блоками имеют сравнительно низкую плотность. Данные блоки называются сообществами, или модулями, и встречаются в разнообразном множестве структур взаимосвязанных объектов.

$L(M)$ – метрика качества (5) полученного разбиения M.

$$L(M) = qH(\mathcal{Q}) + \sum_{i=1}^{m} p_i H(\mathcal{P}^i), \tag{5}$$

где M – разбиение сети на сообщества, m – количество сообществ, $H(\mathcal{Q})$ – энтропия переходов между сообществами, $H(\mathcal{P}^i)$ – энтропия перемещения внутри сообщества i, q – вероятность перехода между сообществами на каждом шаге случайного блуждания, q_i – вероятность покинуть сообщество i, p_α – вероятность посетить вершину α, p_i – вероятность остаться в сообществе i.

Метрика качества полученного разбиения может быть легко подсчитана для любого разбиения, обновление и пересчет этой метрики будет являться быстрой операцией.

Любой численный метод, разработанный для нахождения разбиения сети, который оптимизирует целевую функцию, может быть использован для минимизации метрики $L(M)$.

Отдельным пунктом стоит отметить встроенные алгоритмы авторазмещения графа. Размещение вершин и связей является нетривиальной задачей и в ручном режиме может требовать значительных временных затрат уже для графов с количеством порядка десяти. Поэтому наряду с возможностью ручного размещения вершин и связей предоставляется несколько вариантов автоматического размещения элементов:

- случайное размещение объектов;
- круговое размещение объектов;
- круговое покомпонентное размещение объектов;
- размещение, основанное на алгоритме выделения сообществ;
- размещение «одна линия темы»;
- размещение «две линии темы»;
- размещение, основанное на оценке связности;
- размещение «павлиний хвост».

Размещение, основанное на алгоритме выделения сообществ (рисунок 1), использует полученные сообщества для размещения графа: к каждому из сообществ

может быть применен один из алгоритмов авторазмещений, затем полученные размещения сообществ располагаются по одной окружности так, чтобы не было пересечений размещений вершин из разных сообществ.

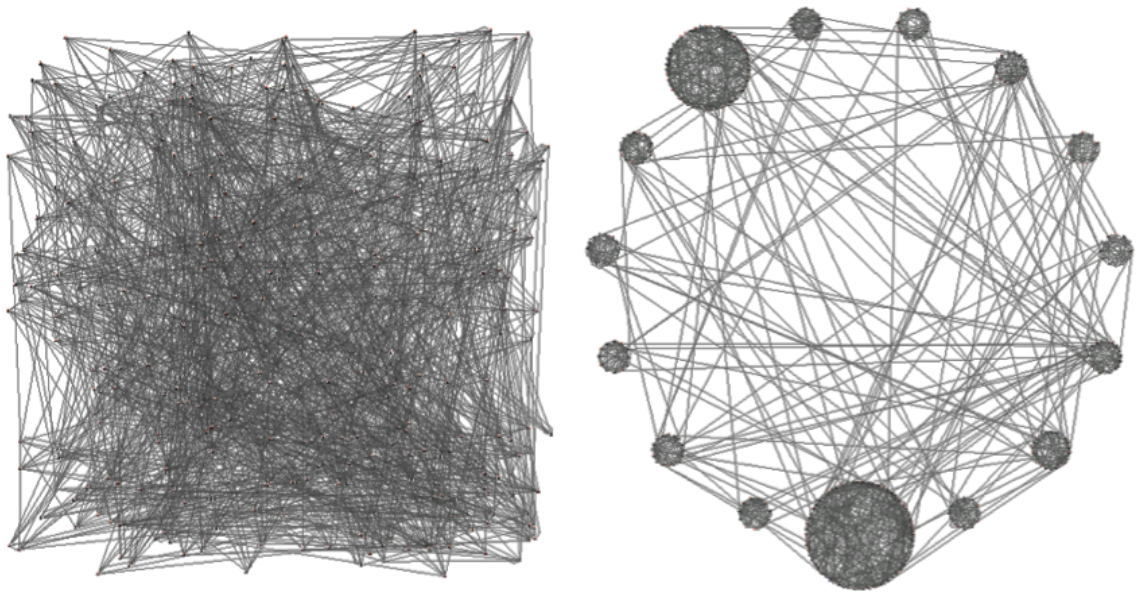

Рис. 1. Пример размещения, основанного на алгоритме выделения сообществ

Размещения «одна линия темы» и «две линии темы» (рисунок 2) базируются на понятии «линия темы». Линия темы – вершина, представленная в виде горизонтальной линии переменной длины с иконкой в любом месте данной линии. Связи с остальными вершинами присоединяются к линии темы перпендикулярно, если второй конец связи попадает в вертикальную полосу, ограниченную концами линии темы, или к одному из концов линии темы в зависимости от того, с какой стороны от данной линии темы расположен второй конец связи. Соответственно данные размещения располагают остальные вершины графа последовательно над и/или под линией темы, минимизируя количество пересечений связей графа между собой. Названия данных размещения говорят сами за себя о количестве используемых линий темы при визуализации всех вершин графа.

Размещение, основывающееся на оценке связности, использует информацию о количестве связей между парами вершин. Условно, перед размещением строится новый граф с тем же количеством вершин и с меньшим количеством связей. Связи отбрасываются по следующему принципу: если количество связей между парой вершин превышает заранее заданный порог, то в новом графе будет присутствовать связь между этими вершинами, в противном случае они не будут связаны. Таким образом, в новом графе появится разбиение на компоненты связности. Полученное разбиение всех вершин исходного графа будет использоваться для его авторазмещения. Вершины всех полученных компонент связности размещаются по отдельности друг от друга с использованием кругового размещения. Затем данные покомпонентные размещения располагаются вдоль горизонтальной прямой, и визуализируются все исходные связи графа.

Размещение «павлиний хвост» (рисунок 3) базируется на основе метода физи-

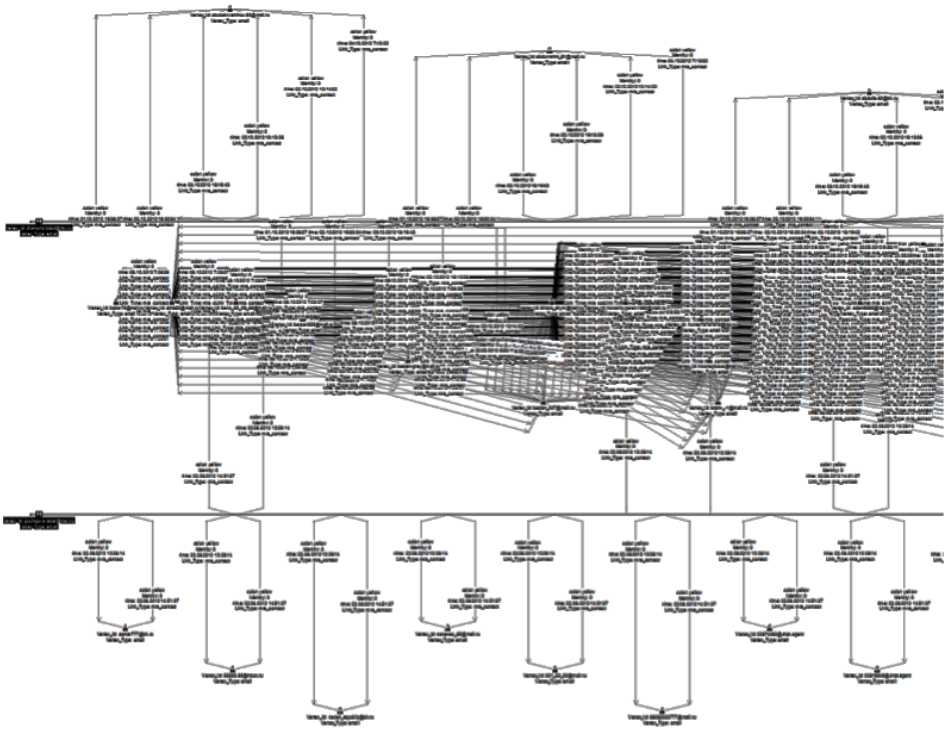

Рис. 2. Пример размещения «две линии темы»

ческих аналогий. Данный подход является очень гибким для размещения графов социальных сетей. Алгоритм вычисляет размещение вершин и связей графа, основываясь лишь на информации, хранящейся в структуре, при этом не принимая во внимание знания предметной области, содержащиеся в вершинах и связях графа.

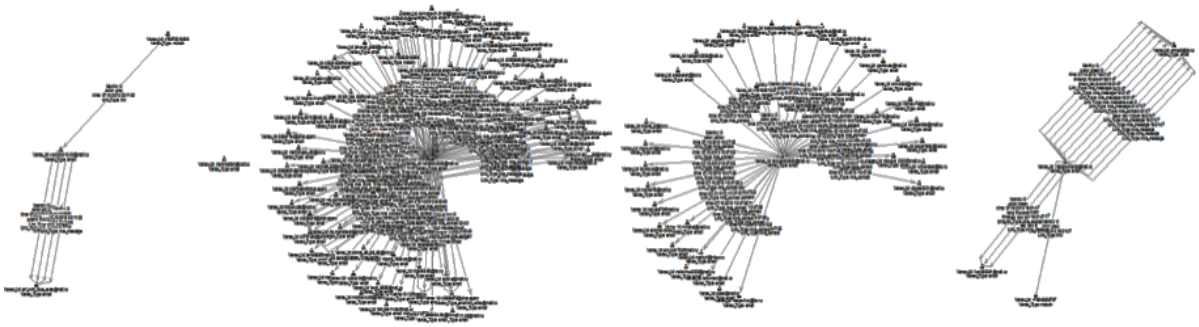

Рис. 3. Пример размещения «павлиний хвост»

Физическая модель строится следующим образом: в качестве вершин рассматриваются одноименно заряженные частицы, взаимодействующие между собой на основе отталкивающей силы взаимодействия между точечными электрическими зарядами по закону Кулона. В качестве связей рассматриваются пружины, таким образом сила притяжения между связанными вершинами описывается законом Гука.

Алгоритм авторазмещения является итерационным. Следовательно, алгоритм

закончит свою работу либо в случае, когда было достигнуто максимальное количество итераций, либо когда динамическая система перешла в состояние равновесия.

Визуализации графов, полученные данным методом, как правило, получаются эстетичными, обладают симметрией и сильно уменьшенным количеством пересечений связей. Базовый метод физических аналогий ограничен размером исследуемого графа: хорошие результаты на небольших графах, плохие результаты на графах, состоящих более чем из сотни вершин. Плохая применимость классического подхода на больших графах обусловлена несколькими причинами. Одной из основных причин является тот факт, что используемая физическая модель обычно имеет большое множество локальных минимумов. Даже использование сложных механизмов получения локального минимума не дает базовому методу хороших результатов при работе на больших графах.

Однако существуют оптимизации, позволяющие получать приемлемые результаты на графах, размеры которых достигают десятки и даже сотни тысяч вершин. Ключевая идея данной оптимизации реализуется в технике многоуровневого размещения, которая заключается в том, чтобы представить граф в более простой форме (сгруппировать вершины в иерархически вложенный набор кластеров), применить классический алгоритм размещения к данному упрощенному представлению и повторить аналогичные операции для каждого из выделенных кластеров. Тем самым базовый метод размещения на основе физических аналогий будет иерархически углубляться от более простых структур к сложным.

Представленные выше авторазмещения позволяют провести детальный анализ сетевой структуры графа.

В данной работе была описана архитектура программного обеспечения по визуализации и анализу больших графов социальных сетей. Представлены описание и анализ структуры хранения графа как в оперативной памяти, так и на жестком диске. Также освещается базовая аналитическая функциональность продукта и возможности модуля визуализации.

В заключение отметим, что рассматриваемый программный комплекс используется в информационных системах для анализа данных различной природы.

Список литературы

1. *Чураков А. Н.* Анализ социальных сетей // СоцИс. 2001. № 1. С. 109–121. [*Churakov A.N.* Analiz sotsial'nykh setey // SotsIs. 2001. No 1. S. 109–121 (in Russian)].

2. *Kaufmann, Wagner.* Drawing Graphs // Springer, 2001. P. 1–274.

3. *Battista, Tamassia, Tollis.* Graph Drawing : Algorithms for the Visualization of Graphs // Springer, 1999. P. 1–430.

4. *Newman M. E.* Networks: An Introduction // Oxford, UK: Oxford University Press, 2010. P. 1–784.

5. i2 Analyst's Notebook. URL: http://www-03.ibm.com/software/products/ru/analysts-notebook. 05.04.2014.

6. Sentinel Visualyzer. URL: http://www.fmsasg.com/Products/SentinelVisualizer. 05.04.2014.

7. CrimeLink. URL: http://www.pciusa.us/Crimelink.aspx. 05.04.2014.

8. Tom Sawyer. URL: http://www.tomsawyer.com. 05.04.2014.

9. XAnalys Link Explorer. URL: http://www.xanalys.com/solutions/linkexplorer.html. 05.04.2014.

Software for Graph Analysis and Visualization

[1]Kolomeychenko M. I., [2]Zolotyh A. A., [1]Polyakov I. V., [1]Chepovskiy A. M.

[1]*National Research University Higher School of Economics, Myasnitskaya str., 20, Moscow, 101000, Russia*
[2]*Moscow State University of Printing Arts, Pryanishnikova str., 20, Moscow, 127550, Russia*

Keywords: graph, graph analysis, graph visualization, graph storage, software

This paper describes the software for graph storage, analysis and visualization. The article presents a comparative analysis of existing software for analysis and visualization of graphs, describes the overall architecture of application and basic principles of construction and operation of the main modules. Furthermore, a description of the developed graph storage oriented to storage and processing of large-scale graphs is presented. The developed algorithm for finding communities and implemented algorithms of autolayouts of graphs are the main functionality of the product. The main advantage of the developed software is high speed processing of large size networks (up to millions of nodes and links). Moreover, the proposed graph storage architecture is unique and has no analogues. The developed approaches and algorithms are optimized for operating with big graphs and have high productivity.

Сведения об авторах:
Коломейченко Максим Игоревич,
Национальный исследовательский университет "Высшая школа экономики",
магистр;
Золотых Алексей Андреевич,
Московский государственный университет печати имени Ивана Федорова,
аспирант;
Поляков Игорь Викторович,
Национальный исследовательский университет "Высшая школа экономики",
канд. физ.-мат. наук;
Чеповский Андрей Михайлович,
Национальный исследовательский университет "Высшая школа экономики",
канд. техн. наук, доцент.